

"Express Mail" Mailing Label No. EL610288115US

PATENT APPLICATION  
ATTORNEY DOCKET NO. SUN-P4431-ARG

5

10 **METHOD AND APPARATUS FOR REACHING  
AGREEMENT BETWEEN NODES IN A  
DISTRIBUTED SYSTEM**

15

**Inventor(s):** Skef F. Iterum and Declan J. Murphy

**Related Application**

20 This application hereby claims priority under 35 U.S.C. § 119 to U. S.  
Provisional Patent Application No. 60/160,992 filed on October 21, 1999, entitled  
"Distributed Multi-Tier Mechanism for Agreement."

**BACKGROUND**

25 **Field of the Invention**

The present invention relates to coordinating activities between nodes in a distributed computing system. More specifically, the present invention relates to a method and an apparatus for reaching agreement between nodes in the distributed computing system regarding a node to function as a primary provider for a  
30 service.

### **Related Art**

As computer networks are increasingly used to link computer systems together, distributed computing systems have been developed to control  
5 interactions between computer systems. Some distributed computing systems allow client computer systems to access resources on server computer systems. For example, a client computer system may be able to access information contained in a database on a server computer system.

When a server computer system fails, it is desirable for the distributed  
10 computing system to automatically recover from this failure. Distributed computer systems possessing an ability to recover from such server failures are referred to as "highly available systems."

For a highly available system to function properly, the highly available system must be able to detect a server failure and reconfigure itself so that  
15 accesses to a failed server are redirected to a backup secondary server.

One problem in designing such a highly available system is that some distributed computing system functions must be centralized in order to operate efficiently. For example, it is desirable to centralize an arbiter that keeps track of where primary and secondary copies of a server are located in a distributed  
20 computing system. However, a node that hosts such a centralized arbiter may itself fail. Hence, it is necessary to provide a mechanism to select a new node to host the centralized arbiter.

Moreover, this selection mechanism must operate in a distributed fashion because, for the reasons stated above, no centralized mechanism is certain to  
25 continue functioning. Furthermore, it is necessary for the node selection process to operate so that the nodes that remain functioning in the distributed computing system agree on the same node to host the centralized arbiter. For efficiency

reasons, it is also desirable for the node selection mechanism not to move the centralized arbiter unless it is necessary to do so.

Hence, what is needed is a method and an apparatus that operates in a distributed manner to select a node to host a primary server for a service.

5

### SUMMARY

One embodiment of the present invention provides a system for selecting a node to host a primary server for a service from a plurality of nodes in a distributed computing system. The system operates by receiving an indication that a state of the distributed computing system has changed. In response to this indication, the system determines if there is already a node hosting the primary server for the service. If not, the system selects a node to host the primary server using the assumption that a given node from the plurality of nodes in the distributed computing system hosts the primary server. The system then communicates rank information between the given node and other nodes in the distributed computing system, wherein each node in the distributed computing system has a unique rank with respect to the other nodes in the distributed computing system. The system next compares the rank of the given node with the rank of the other nodes in the distributed computing system. If one of the other nodes has a higher rank than the given node, the system disqualifies the given node from hosting the primary server.

In one embodiment of the present invention, if there exists a node to host the primary server, the system allows the node that hosts the primary server to communicate with other nodes in the distributed computing system in order to disqualify the other nodes from hosting the primary server.

In one embodiment of the present invention, the system maintains a candidate variable in the given node identifying a candidate node to host the

primary server. In a variation on this embodiment, the system initially sets the candidate variable to identify the given node.

In one embodiment of the present invention, after a new node has been selected to host the primary server, if the new node is different from a previous  
5 node that hosted the primary server, the system maps connections for the service to the new node. In a variation on this embodiment, the system also configures the new node to host the primary server for the service.

In one embodiment of the present invention, the system restarts the service if the service was interrupted as a result of the change in state of the distributed  
10 computing system.

In one embodiment of the present invention, the given node in the distributed computing system can act as one of: a host for the primary server for the service; a host for a secondary server for the service, wherein the secondary server periodically receives checkpointing information from the primary server; or  
15 a spare for the primary server, wherein the spare does not receive checkpointing information from the primary server.

In one embodiment of the present invention, upon initial startup of the service, the system selects a highest ranking spare to host the primary server for the service.

In one embodiment of the present invention, the system allows the primary  
20 server to configure spares in the distributed computing system to host secondary servers for the service.

In one embodiment of the present invention, comparing the rank of the given node with the rank of the other nodes in the distributed computing system  
25 involves considering a host for a secondary server to have a higher rank than a spare.

In one embodiment of the present invention, after disqualifying the given node from hosting the primary server, the system ceases to communicate rank information between the given node and the other nodes in the distributed computing system.

5

## BRIEF DESCRIPTION OF THE FIGURES

FIG. 1 illustrates a distributed computing system in accordance with an embodiment of the present invention.

FIG. 2 illustrates how highly available services are controlled within a distributed computing system in accordance with an embodiment of the present invention.

FIG. 3 illustrates how a replica managers controls highly available services in accordance with an embodiment of the present invention.

FIG. 4 is a flow chart illustrating the process of selecting and configuring a new primary server in accordance with an embodiment of the present invention.

FIG. 5 is a flow chart illustrating some of the operations performed by a primary server in accordance with an embodiment of the present invention.

FIG. 6 illustrates how a node is selected to host a primary server through a disqualification process in accordance with an embodiment of the present invention.

FIG. 7 illustrates how nodes a disqualified in accordance with an embodiment of the present invention.

## DETAILED DESCRIPTION

The following description is presented to enable any person skilled in the art to make and use the invention, and is provided in the context of a particular application and its requirements. Various modifications to the disclosed

embodiments will be readily apparent to those skilled in the art, and the general principles defined herein may be applied to other embodiments and applications without departing from the spirit and scope of the present invention. Thus, the present invention is not intended to be limited to the embodiments shown, but is  
5 to be accorded the widest scope consistent with the principles and features disclosed herein.

The data structures and code described in this detailed description are typically stored on a computer readable storage medium, which may be any device or medium that can store code and/or data for use by a computer system. This  
10 includes, but is not limited to, magnetic and optical storage devices such as disk drives, magnetic tape, CDs (compact discs) and DVDs (digital versatile discs or digital video discs), and computer instruction signals embodied in a transmission medium (with or without a carrier wave upon which the signals are modulated). For example, the transmission medium may include a communications network,  
15 such as the Internet.

### **Distributed Computing System**

FIG. 1 illustrates a distributed computing system 100 in accordance with an embodiment of the present invention. Distributed computing system 100  
20 includes a number of computing nodes 102-105, which are coupled together through a network 110.

Network 110 can include any type of wire or wireless communication channel capable of coupling together computing nodes. This includes, but is not limited to, a local area network, a wide area network, or a combination of  
25 networks. In one embodiment of the present invention, network 110 includes the Internet. In another embodiment of the present invention, network 110 is a local

high speed network that enables distributed computing system 100 to function as a clustered computing system (hereinafter referred to as a "cluster").

Nodes 102-105 can generally include any type of computer system, including, but not limited to, a computer system based on a microprocessor, a  
5 mainframe computer, a digital signal processor, a personal organizer, a device controller, and a computational engine within an appliance.

Nodes 102-105 also host servers, which include a mechanism for servicing requests from a client for computational and/or data storage resources. More specifically, node 102 hosts primary server 106, which services requests from  
10 clients (not shown) for a service involving computational and/or data storage resources.

Nodes 103-104 host secondary servers 107-108, respectively, for the same service. These secondary servers act as backup servers for primary server 106. To this end, secondary servers 107-108 receive periodic checkpoints 120-121  
15 from primary server 106. These periodic checkpoints enable secondary servers 107-108 to maintain consistent state with primary server 106. This makes it possible for one of secondary servers 107-108 to take over for primary server 106 if primary server 106 fails.

Node 105 can serve as a spare node to host the service provided by  
20 primary server 106. Hence, node 105 can be configured to host a secondary server with respect to a service provided by primary server 106. Alternatively, if all primary servers and secondary servers for the service fail, node 105 can be configured to host a new primary server for the service.

Also note that nodes 102-105 contain distributed selection mechanisms  
25 132-135, respectively. Distributed selection mechanisms 132-135 communicate with each other to select a new node to host primary server 106 when node 102 fails or otherwise becomes unavailable. This process is described in more detail

below with reference to FIGs. 2-6.

### **Controlling Highly Available Services**

FIG. 2 illustrates how highly available services 202-205 are controlled  
5 within distributed computing system 100 in accordance with an embodiment of  
the present invention. Note that highly available services 202-205 continue to  
operate even if individual nodes of distributed computing system 100 fail.

Highly available services 202-205 operate under control of replica  
manager 206. Referring to FIG. 3, for each service, replica manager 206 keeps a  
10 record of which nodes in distributed computing system 100 function as primary  
servers, and which nodes function as secondary servers. For example, in FIG. 3  
replica manager 206 keeps track of highly available services 202-205. The  
primary server for service 202 is node 103, and the secondary servers are nodes  
104 and 105. The primary server for service 203 is node 104, and the secondary  
15 servers are nodes 103 and 105. The primary server for service 204 is node 102,  
and the secondary servers are nodes 104-105. The primary server for service 205  
is node 103, and the secondary servers are nodes 102, 104 and 105.

Replica manager 206 additionally performs a number of related functions,  
such as configuring a node to host a primary (which may involve demoting a  
20 current host-for the primary to host a secondary). Replica manager 206 may  
additionally perform other functions, such as: adding a service; removing  
providers for a service; registering providers for a service; removing a service;  
handling provider failures; bringing up new providers for a service; and handling  
dependencies between services (which may involve ensuring that primaries for  
25 dependent services are co-located on the same node).



Referring back to FIG. 2, replica manager 206 is itself a highly available service that operates under control of replica manager manager (RMM) 208. Note that RMM 208 is not managed by a higher level service.

As illustrated in FIG. 2, RMM 208 communicates with cluster membership monitor (CMM) 210. CMM 210 monitors cluster membership and alerts RMM 208 if any changes in the cluster membership occur.

CMM 210 uses transport layer 212 to exchange messages between nodes 102-105.

#### 10 **Process of Selecting a New Primary**

FIG. 4 is a flow chart illustrating the process of selecting and configuring a primary server in accordance with an embodiment of the present invention. Note that this process is run concurrently by each active node in distributed computing system 100. The system begins by receiving an indication from CMM 210 that the membership in the cluster has changed (step 401).

In response to this indication, the system obtains a lock on a local candidate variable which contains an identifier for a candidate node to host primary server 106 (step 402). The system also obtains an additional lock to hold off requesters for the service (step 404).

Next, the system executes a disqualification process by communicating with other nodes in distributed computing system 100 in order to disqualify the other nodes from acting as the primary server 106 (step 406). This process is described in more detail with reference to FIG. 6 below.

After the disqualification process, the remaining node, which is not disqualified, becomes the primary node. If the node hosting primary server 106 has changed, this may involve re-mapping connections for the service to point to

the new node (step 408). It may also involve initializing the new node to act as the host for the primary (step 410).

Finally, the service is started (step 412). This may involve unfreezing the service if it was previously frozen, as well as releasing the previously obtained  
5 lock that holds off requesters for the service.

FIG. 5 is a flow chart illustrating some of the operations performed by a primary server 106 in accordance with an embodiment of the present invention. During operation, primary server 106 performs periodic checkpointing operations 120-121 with secondary servers 107-108, respectively (step 502). These  
10 checkpointing operations allow secondary servers 107-108 to take over from primary server 106 if primary server 106 fails. Primary server 106 also periodically attempts to promote spare nodes (such as node 105 in FIG. 1) to host secondaries (step 504). This promotion process involves transferring state information to a spare node in order to bring the spare node up to date with  
15 respect to the existing secondaries.

FIG. 6 illustrates how a node is selected to host primary server 106 through a disqualification process in accordance with an embodiment of the present invention. Note that FIG. 6 describes in more detail the process described above with reference to step 406 in FIG. 4.

20 The system starts by determining if a node that was previously hosting primary server 106 continues to exist (step 602).

If not, the system retrieves the state of a local provider for the service (step 604). The system then sets the candidate variable to identify the local provider (step 606), and subsequently unlocks the candidate lock that was set previously in  
25 step 402 of FIG. 4 (step 608). Next, if the candidate is not the local provider, the system ends the process (step 610).

Next, for all other nodes I in the cluster, the system attempts to disqualify node I by writing a new identifier into the candidate variable for node I if the rank of node I is less than the rank of the present node (step 612). This process is described in more detail with reference to FIG. 7 below. Finally, if node I's local  
5 provider has a higher rank than the present node, the process terminates because the present node is disqualified (step 614).

Note that a rank of a node can be obtained by comparing a unique identifier for the node with unique identifiers for other nodes. Also note that the rank of a primary server is greater than the rank of a secondary server, and that the  
10 rank of a secondary server is greater than the rank of a spare. The above-listed restrictions on rank ensure that an existing primary that has not failed continues to function as the primary, and that an existing secondary will be chosen ahead of a spare. Of course, when the system is initialized, no primaries or secondaries exist, so a spare is selected to be the primary.

On the other hand, if the node that was hosting primary server 106  
15 continues to function, the system sets the candidate to be this node (step 616), and unlocks the candidate node (step 618).

If the present node does not host primary server 106, the process is finished. Otherwise, if the present node is hosting primary server 106, the system  
20 considers each other node I in the cluster. If the present node has already communicated with I, the system skips node I (step 622). Otherwise, the system communicates with node I in order to disqualify node I from acting as the host for primary server 106 (step 624). This may involve causing an identifier for the present node to be written into the candidate variable for node I.

FIG. 7 illustrates how nodes are disqualified in accordance with an  
25 embodiment of the present invention. Note that FIG. 7 describes in more detail the process described above with reference to step 612 in FIG. 6. The caller first

